**3.1. Google Maps**

API Google Карт предоставляет веб-службы в качестве интерфейса для запроса данных API Карт из внешних служб и использования этих сведений в приложениях на базе Карт. Согласно условиям [Ограничения лицензии на использование API Google Карт](https://developers.google.com/maps/terms?hl=ru#section_10_12) эти службы не могут использоваться отдельно от карт Google.[7]

Данные веб-службы используют HTTP-запросы к определенным URL, передавая параметры URL в качестве аргументов. Обычно эти службы возвращают приложению HTTP-запрос, содержащий данные в формате JSON или XML для их синтаксического анализа и обработки.

Запрос к веб-службе обычно имеет следующий вид:

http://maps.googleapis.com/maps/api/*service*/*output*?*parameters*

где service указывает запрашиваемую службу, а output – формат ответа (обычно json или xml).

Доступ к веб-службам API Google Карт можно также осуществлять по протоколу **HTTPS.**  Для этого следует изменить протокол в URL запроса на https, как показано ниже:

**https**://maps.googleapis.com/maps/api/service/output?parameters

Запросы от приложений, работающих с конфиденциальными данными пользователей, например, с данными об их местоположении, рекомендуется отправлять по протоколу HTTPS.

Для использования Google JS API в браузере нужно подключить его код:

*<script src="https://maps.googleapis.com/maps/api/js?key=YOUR\_API\_KEY&sensor=SET\_TO\_TRUE\_OR\_FALSE" type="text/javascript"></script>*

С помощью объекта *DirectionsService* можно рассчитывать маршруты для различных способов передвижения. Этот объект взаимодействует со службой маршрутов интерфейса API Google Карт, которая получает запрос маршрута и возвращает вычисленные результаты.

В службе маршрутов пункты отправления и назначения могут указываться в виде текстовых запросов (например, "Чикаго, Иллинойс, США" или "Дарвин, Новый Южный Уэльс, Австралия") либо в виде координат LatLng. Результаты возвращаются в виде последовательности отрезков, проходящих через путевые точки. Маршруты отображаются в виде полилинии, показывающей маршрут на карте, или дополнительно в виде последовательности текстовых описаний в элементе <div> (например, "Поверните направо для въезда на Троицкий мост").

Чтобы использовать маршруты в версии 3, нужно создать объект типа *DirectionsService* и вызвать метод *DirectionsService.route()*для отправки запроса в службу маршрутов, передавая ей литерал объекта*DirectionsRequest*, содержащий условия ввода и метод обратного вызова для выполнения после получения ответа.

Литерал объекта *DirectionsRequest* содержит следующие поля:

*{  
  origin: LatLng | String,  
  destination: LatLng | String,  
  travelMode: TravelMode,  
  transitOptions: TransitOptions,  
  unitSystem: UnitSystem,  
  waypoints[]: DirectionsWaypoint,  
  optimizeWaypoints: Boolean,  
  provideRouteAlternatives: Boolean,  
  avoidHighways: Boolean,  
  avoidTolls: Boolean  
  region: String  
}*

Далее идет описание этих полей:

origin (обязательный параметр) указывает начальное местоположение, от которого следует вычислять маршрут. Этот параметр может иметь значение типа String (например, "Чикаго, Иллинойс, США") или LatLng.

destination (обязательный параметр) указывает конечное местоположение, до которого следует вычислять маршрут. Этот параметр может иметь значение типа String (например, "Чикаго, Иллинойс, США") или LatLng.

travelMode (обязательный параметр) задает способ перемещения, который используется для вычисления маршрута. Допустимые значения указаны далее в разделе [Способы перемещения](https://developers.google.com/maps/documentation/javascript/directions?hl=ru#TravelModes).

transitOptions (необязательный параметр) указывает значения, используемые только в запросах, в которых параметрtravelMode имеет значение google.maps.TravelMode.TRANSIT. Допустимые значения описаны далее в разделе [Параметры маршрутов общественного транспорта](https://developers.google.com/maps/documentation/javascript/directions?hl=ru#TransitOptions).

unitSystem (необязательный параметр) задает единицы измерения, которые следует использовать при отображении результатов. Допустимые значения указаны далее в разделе [Системы измерений](https://developers.google.com/maps/documentation/javascript/directions?hl=ru#UnitSystems).

waypoints[] (необязательный параметр) указывает массив элементов DirectionsWaypoint. Использование путевых точек приводит к изменению маршрута, направляя его через один или несколько указанных пунктов. Путевая точка задается как литерал объекта со следующими полями:

location задает местоположение путевой точки, которая будет геокодирована, как LatLng или String.

stopover содержит логическое значение, указывающее, что путевая точка является остановкой на маршруте, что приводит к его разделению на две части.

Дополнительные сведения о путевых точках см. ниже в разделе [Использование путевых точек в маршрутах](https://developers.google.com/maps/documentation/javascript/directions?hl=ru#Waypoints).

optimizeWaypoints (необязательный параметр) указывает, что с помощью предоставленных элементов waypoints маршрут можно оптимизировать для получения кратчайшего пути. Если в этом поле указано значение true, служба маршрутов возвращает переупорядоченные элементы waypoints в поле waypoint\_order.Дополнительные сведения см. далее в разделе [Использование путевых точек в маршрутах](https://developers.google.com/maps/documentation/javascript/directions?hl=ru#Waypoints).

provideRouteAlternatives (необязательный параметр) – значение true указывает, что служба может предложить несколько альтернативных маршрутов. Обратите внимание, что расчет альтернативных путей может увеличить время отклика сервера.

avoidHighways (необязательный параметр) – значение true указывает, что проложенные маршруты должны по возможности избегать автомагистралей.

avoidTolls (необязательный параметр) – значение true указывает, что проложенные маршруты должны по возможности избегать платных дорог.

region (необязательный параметр) указывает код региона в виде двухсимвольного значения ccTLD (домен верхнего уровня). Дополнительные сведения см. далее в разделе [Предпочитаемый регион](https://developers.google.com/maps/documentation/javascript/directions?hl=ru#DirectionsRegionBiasing).

**Статус запроса маршрута**

*DirectionsStatus*может возвратить следующие значения:

*OK* указывает, что ответ содержит допустимый элемент DirectionsResult.

*NOT\_FOUND* означает, что по крайней мере для одного указанного пункта (исходный, пункт назначения или путевая точка) не удалось выполнить геокодирование.

*ZERO\_RESULTS* означает, что между исходной точкой и пунктом назначения не найдено ни одного маршрута.

*MAX\_WAYPOINTS\_EXCEEDED* указывает, что в *DirectionsRequest* задано слишком много объектов *DirectionsWaypoint*. Максимальное разрешенное количество путевых точек составляет 8 помимо исходной точки и пункта назначения. Клиентам API Google Карт для организаций разрешено использовать 23 путевые точки помимо исходной точки и пункта назначения. Для маршрутов общественного транспорта путевые точки не поддерживаются.

*INVALID\_REQUEST* указывает, что был предоставлен недопустимый объект*DirectionsRequest*. В большинстве случаев этот код ошибки возвращается для запросов без исходной точки и пункта назначения, а также для запросов маршрутов общественного транспорта с путевыми точками.

*OVER\_QUERY\_LIMIT* означает, что в разрешенный период времени веб-страница отправила слишком много запросов.

*REQUEST\_DENIED* означает, что веб-странице не разрешено использовать службу маршрутов.

*UNKNOWN\_ERROR* означает, что обработка запроса маршрута невозможна из-за ошибки сервера. При повторной попытке запрос может быть успешно выполнен.

**Объект DirectionsResult**

При отправке запроса маршрута в *DirectionsService* вы получите ответ, состоящий из кода статуса и результата, которым является объект *DirectionsResult*. *DirectionsResult* представляет собой литерал объекта с единственным полем:

*routes[]* содержит массив объектов *DirectionsRoute*. Каждый маршрут указывает путь от исходной точки до пункта назначения, предоставленный в объекте *DirectionsRequest*. Обычно для любого запроса возвращается только один маршрут, за исключением случаев, когда для поля *provideRouteAlternatives*запроса установлено значение true. В таких случаях может быть возвращено несколько маршрутов.

**Ограничения на использование**

На использование службы API маршрутов Google накладывается ограничение, составляющее 2500 запросов маршрутов в день. Поиск автомобильного, велосипедного или пешеходного маршрута учитывается при расчете ежедневной квоты как один запрос. Поиск маршрута общественного транспорта учитывается как 4 запроса.

Отдельные запросы автомобильных, велосипедных или пешеходных маршрутов могут содержать до 8 промежуточных путевых точек. Пользователи [API Google Карт для организаций](https://developers.google.com/maps/documentation/business?hl=ru) в течение дня могут запрашивать до 100 000 маршрутов, для каждого из которых допускается до 23 путевых точек. Для маршрутов общественного транспорта путевые точки не поддерживаются.

Кроме того, прежде чем выполнять [кодирование адресов URL](https://developers.google.com/maps/documentation/webservices/index?hl=ru#BuildingURLs), убедитесь в том, что адреса URL для API маршрутов содержат не более 2048 символов. Помните об этом ограничении, формируя адреса URL, поскольку некоторые адреса URL, используемые службой маршрутов, могут привести к включению в маршрут нескольких пунктов.

Примечание. API маршрутов можно применять только для отображения результатов на карте Google. Использование данных службы маршрутов без отображения карты, для которой они запрашивались, запрещено. Кроме того, при вычислении маршрутов формируются данные об авторских правах и предупреждения, которые каким-либо способом должны быть показаны пользователю. Полную информацию о допустимых способах применения см. в разделе [Ограничения лицензии на использование API Карт](https://developers.google.com/maps/terms?hl=ru#section_10_12).

**API высотных данных**

API высотных данных предоставляет данные о высотах всех точек на земной поверхности, включая глубины океана (в этом случае возвращаются отрицательные значения). Если у Google нет точных высотных данных для интересующей вас точки, служба использует интерполяцию и возвращает усредненное значение по четырем ближайшим точкам[7].

С помощью службы API высотных данных можно разрабатывать приложения для пешего и велосипедного туризма, приложения мобильного позиционирования или геодезические приложения низкого разрешения.

Доступ к API высотных данных осуществляется через интерфейс HTTP. Пользователи JavaScript API Google версии 3 могут также напрямую обращаться к API высотных данных с помощью объекта ElevationService().

На использование API высотных данных Google накладываются ограничения в отношении количества запросов: не более 2500 в день (для пользователей API Карт для организаций – не более 100 000 в день). В каждом запросе разрешается указывать до 512 местоположений, но не более 25 000 в день (для пользователей API Карт для организаций – 1 000 000). Если превышено суточное (24 часа) ограничение или имеет место злоупотребление другого рода, API высотных данных может временно прекратить работу. При неоднократных нарушениях доступ к нему может быть заблокирован.

Для запросов по одной точке API высотных данных возвращает данные с максимальной точностью. Для пакетных запросов по нескольким точкам могут быть возвращены менее точные данные, особенно если эти точки удалены друг от друга, так как выполняется сглаживание данных.

URL для API высотных данных может содержать не более 2048 символов.

API высотных данных можно использовать только для отображения результатов на карте Google. Использование данных службы высотных данных без отображения карты, для которой они запрашивались, запрещено..

**Запросы к службе высотных данных**

API высотных данных возвращает данные о высоте в разных точках на поверхности земли. Указать требуемые точки можно одним из следующих способов:

В виде набора из одного или нескольких местоположений (элементы locations).

В виде последовательности соединенных точек вдоль пути (элемент path).

Каждый из этих методов использует значения широты и долготы для определения местоположения или вершин пути. URL для API высотных данных должен иметь следующий формат:

http://maps.googleapis.com/maps/api/elevation/*outputFormat*?*parameters*

Для доступа к API высотных данных можно также использовать SSL, заменив https на http в URL запроса. Запросы от приложений, работающих с конфиденциальными данными пользователей, например, с данными об их местоположении, рекомендуется отправлять по протоколу HTTPS.

**Форматы вывода**

Форматы вывода указываются в URL запроса с помощью специального флага службы. В настоящее время API высотных данных поддерживает следующие форматы вывода:

/json – результаты возвращаются в формате [JavaScript Object Notation](http://en.wikipedia.org/wiki/JSON) (JSON).

/xml – результаты возвращаются в формате XML, заключенными в узел <ElevationResponse>.

**Ответы на запросы к службе высотных данных**

Для каждого допустимого запроса служба высотных данных возвращает ответ в формате, указанном в URL запроса. Каждый ответ содержит следующие элементы:

**Код состояния высоты (**status**)**, который может иметь одно из следующих значений:

OK, означающее, что запрос к API успешно выполнен.

INVALID\_REQUEST, означающее, что запрос к API сформирован неправильно.

OVER\_QUERY\_LIMIT, означающее, что отправитель запроса превысил квоту.

REQUEST\_DENIED, означающее, что API не выполнил запрос, вероятнее всего, по той причине, что отправитель запроса не указал допустимое значение параметра sensor).

UNKNOWN\_ERROR, обозначающее неизвестную ошибку.

Массив results, содержащий следующие элементы:

Элемент location (содержащий элементы lat и lng) той точки, для которой вычисляются данные о высоте. Следует отметить, что для запросов пути набор элементов location будет содержать выборочные точки вдоль пути.

Элемент elevation, указывающий высоту точки над уровнем моря в метрах.

Значение resolution, указывающее максимальное расстояние (в метрах) между точками данных, на основании которых интерполируется значение высоты. Если разрешение неизвестно, это свойство отсутствует.

**Примеры высотных данных для отдельных точек**

В следующем примере запрашивается высота над уровнем моря Денвера, штат Колорадо, в формате JSON (этот город известен как "город высотой в милю", так как находится на отметке 1 мили над уровнем моря):

*http://maps.googleapis.com/maps/api/elevation/json?locations=39.7391536,-104.9847034&sensor=true\_or\_false*

*{  
   "results" : [  
      {  
         "elevation" : 1608.637939453125,  
         "location" : {  
            "lat" : 39.73915360,  
            "lng" : -104.98470340  
         },  
         "resolution" : 4.771975994110107  
      }  
   ],  
   "status" : "OK"  
}*

**Пример запроса и ответа для Денвера, штат Колорадо, и Долины смерти, штат Калифорния):**

*http://maps.googleapis.com/maps/api/elevation/json?locations=39.7391536,-104.9847034|36.455556,-116.866667&sensor=true\_or\_false*

*{  
   "results" : [  
      {  
         "elevation" : 1608.637939453125,  
         "location" : {  
            "lat" : 39.73915360,  
            "lng" : -104.98470340  
         },  
         "resolution" : 4.771975994110107  
      },  
      {  
         "elevation" : -50.78903579711914,  
         "location" : {  
            "lat" : 36.4555560,  
            "lng" : -116.8666670  
         },  
         "resolution" : 19.08790397644043  
      }  
   ],  
   "status" : "OK"  
}*

### *Пример запроса высотных данных пути*

*http://maps.googleapis.com/maps/api/elevation/json?path=36.578581,-118.291994|36.23998,-116.83171&samples=3&sensor=true\_or\_false*

**3.2. Яндекс Карты**

**Условия использования**

* API Яндекс.Карт предназначен для отображения карты реальным пользователям. Нельзя применять его только для получения и обработки данных, например геокодирования точек или прокладки маршрутов без вывода их в интерфейсе.
* Недопустимо скрывать или менять логотипы и копирайты на карте, а также загораживать их другими элементами.
* Бесплатный API Яндекс.Карт можно использовать только на общедоступных сайтах и в бесплатных приложениях. У сервисов с авторизацией должна быть открытая и бесплатная регистрация.
* Бесплатный API Карт нельзя использовать в модулях бухгалтерских программ, интранетах и других закрытых системах, десктоп-приложениях и программах, игровых проектах, а также системах мониторинга и диспетчеризации транспорта.
* Также в бесплатном API есть ограничение — в сутки можно делать не более 25 000 запросов к геокодеру.[7]

## Загрузка модулей при подключении API

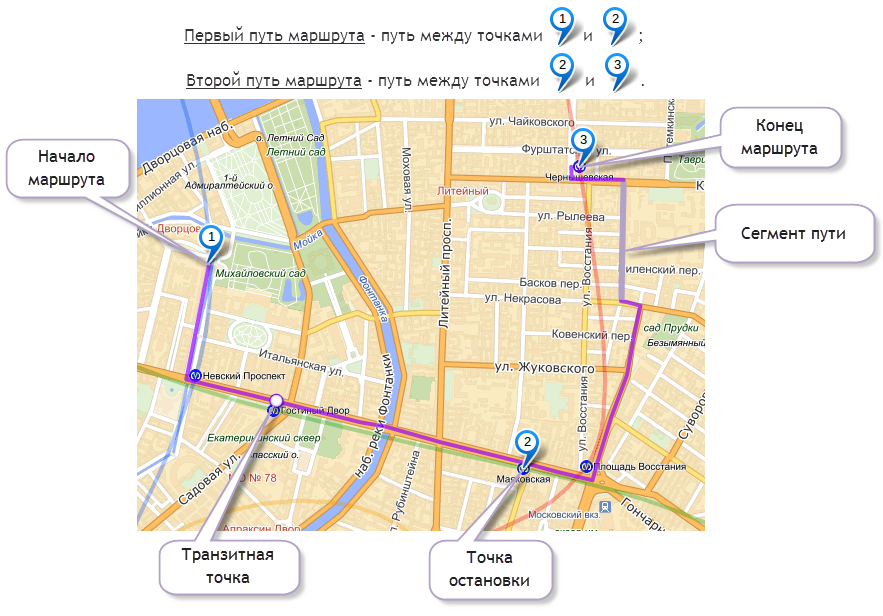
Загружаемые модули задаются HTTP-параметром load в строке подключения API. Можно указать сразу несколько модулей, разделив их запятым. По умолчанию параметр load принимает значениеpackage.full, то есть загружаются все основные модули, необходимые для работы API.

*<script src="//api-maps.yandex.ru/2.1/?lang=ru\_RU&****load****=Map,Placemark" type="text/javascript"></script>*

Загруженные модули будут включены в публичное пространство имен ymaps.

# Маршрутизатор

API предоставляет возможность прокладывания автомобильных маршрутов. Маршрут между начальным и конечным пунктом вычисляется автоматически, при этом можно задать произвольное количество точек остановки и транзитных точек маршрута.



Для построения маршрута предназначена функция [route](https://tech.yandex.ru/maps/doc/jsapi/2.1/ref/reference/route-docpage/), в которую передается массив точек, через которые нужно проложить маршрут и, при необходимости, дополнительные опции его построения.

Вычисление маршрута производится асинхронно (аналогично [геокодированию](https://tech.yandex.ru/maps/doc/jsapi/2.1/dg/concepts/geocoding-docpage/#async)). Результат передается в функцию-обработчик в виде коллекции [*GeoObjectCollection*](https://tech.yandex.ru/maps/doc/jsapi/2.1/ref/reference/GeoObjectCollection-docpage/). Этот объект реализует интерфейс [*IGeoObject*](https://tech.yandex.ru/maps/doc/jsapi/2.1/ref/reference/IGeoObject-docpage/), то есть может быть размещен на карте.

*ymaps.route(['Москва', 'Санкт-Петербург']).then(*

*function (route) {*

*myMap.geoObjects.add(route);*

*},*

*function (error) {*

*alert('Возникла ошибка: ' + error.message);*

*}*

*);*

Существует три способа задания точки маршрута:

Пара координат в виде массива ([59.94, 30.31]).

Строка ('Санкт-Петербург'). Производится автоматическое [геокодирование](https://tech.yandex.ru/maps/doc/jsapi/2.1/dg/concepts/geocoding-docpage/).

Объект с полями *type* и *point*

*({type: 'wayPoint', point: 'Санкт-Петербург'}, {type: 'viaPoint', point: [59.94, 30.31]).*

Этот способ позволяет в явном виде указать тип точки маршрута: *wayPoint* — точка остановки, *viaPoint* — транзитная точка. Если в поле *point* передана строка, производится автоматическое геокодирование.

Если тип точки не указан, она считается точкой остановки.

*ymaps.route([*

*'Кронштадт, Якорная площадь',*

*{*

*type: 'viaPoint',*

*point: [59.93328,30.342791]// или 'Аничков мост'*

*},*

*'Санкт-Петербург, Финляндский вокзал'// или [59.956084, 30.356849]*

*]).then(*

*function (route) {*

*myMap.geoObjects.add(route);*

*},*

*function (error) {*

*alert("Возникла ошибка: " + error.message);*

*}*

*);*

Прокладывать маршрут можно как с учетом так и без учета пробок. Игнорирование или учет пробок регулируется опцией avoidTrafficJams (true/false). По умолчанию пробки не учитываются. Маршрут, проложенный без учета пробок, может отличаться от маршрута, проложенного между теми же точками, но с учетом пробок.

*ymaps.route(['Кронштадт, Якорная площадь', 'Санкт-Петербург, Финляндский вокзал'], {****avoidTrafficJams: true****});*

Построенный маршрут представляется в виде [упорядоченной коллекции](https://tech.yandex.ru/maps/doc/jsapi/2.1/ref/reference/GeoObjectCollection-docpage/) путей ([router.Path](https://tech.yandex.ru/maps/doc/jsapi/2.1/ref/reference/router.Path-docpage/)), попарно последовательно соединяющих точки остановки. Получить пути маршрута можно с помощью метода[getPaths](https://tech.yandex.ru/maps/doc/jsapi/2.1/ref/reference/router.Route-docpage/#getPaths). Для предыдущего примера с промежуточной точкой «Аничков мост» кодroute.getPaths().getLength() возвратит единицу, так как промежуточные точки не влияют на разбиение маршрута на пути.

Каждый путь разбивается на сегменты ([router.Segment](https://tech.yandex.ru/maps/doc/jsapi/2.1/ref/reference/router.Segment-docpage/)), получить которые из пути можно с помощью метода [getSegments](https://tech.yandex.ru/maps/doc/jsapi/2.1/ref/reference/router.Path-docpage/#getSegments). Концами сегментов являются начальные и конечные точки путей, транзитные точки, точки вынужденной остановки (например, паром) и точки возможного изменения направления движения (развилка, въезд, съезд, поворот, разворот, перекресток).

Сегмент описывается ломаной, координаты точек ломаной возвращаются методом [getCoordinates](https://tech.yandex.ru/maps/doc/jsapi/2.1/ref/reference/router.Segment-docpage/#getCoordinates).

*var routeLength = route.getLength(); // Длина маршрута*

*var firstPath = route.getPaths().get(0); // Первый путь*

*var firstPathLength = firstPath.getLength(); // Длина первого пути*

*var firtstPathTime = firstPath.getTime(); // Время без учета пробок*

*var firstPathFirstSegment = firstPath.getSegments()[0];// Первый сегмент первого пути*

*var firstPathFirstSegmentJamsTime = firstPathFirstSegment.getJamsTime();//* Время с пробками

Для сегментов также доступны данные, позволяющие построить маршрутный лист: направление и угол поворота в конце сегмента, а также название улицы, на которой он начинается.

Сервиса высотных данных у Яндекс нет.